Capitolul 7. Proiectarea conceptuală

- Scop: reprezentarea cerinţelor informale ale aplicaţiei în termenii descrierii complete şi formale dar independent de criteriul folosit pentru reprezentare în sistemul de management al bazei de date.

- Rezultat: *schema conceptuală* - *model de date conceptual* - permite descrierea organizării datelor la un nivel înalt de abstractizare fără a lua în considerare aspectele de implementare.

- Proiectarea conceptuală a bazelor de date constă în construirea unei scheme Entitate-Relaţie care furnizează o descriere optimă a cerinţelor clienţilor.

- Construcţia schemei este un proces iterativ, aceasta suferind o serie de transformări şi corecţii.

- În acest capitol se vor descrie strategii pentru dezvoltarea unei scheme conceptuale.

7.1 Extragerea şi analiza cerinţelor

Extragerea cerinţelor - identificarea completă a problemelor pe care aplicaţia trebuie să le rezolve şi a caracteristicilor aplicaţiei.

Cerinţele sunt transformate în specificaţii care în general sunt exprimate în limbaj natural şi din acest motiv pot fi ambigue şi dezorganizate.

Analiza cerinţelor - clarificarea şi organizarea specificaţiilor cerinţelor.

Cerinţele pot proveni din mai multe surse, cum ar fi:

- *Utilizatori ai aplicaţiei -* informaţia este obţinută prin interviuri sau prin intermediul unor documente specifice scrise special pentru acest scop.
- *Documentaţie existentă referitoare la problema de rezolvat* - reguli interne, proceduri de operare etc. Sunt necesare colectarea şi selecţia. Responsabilitatea revine proiectantului.
- *Posibile aplicaţii anterioare* care trebuie să fie înlocuite sau cu care noua aplicaţie trebuie să interacţioneze.

Exemplu. Se cere proiectarea unei baze de date pentru o companie de training şi pentru care s-au colectat specificaţiile prezentate în tabelul următor. Datele au fost extrase prin interviuri cu angajaţii companiei.

1	Dorim crearea unei baze de date pentru o companie care face cursuri de instruire
2	Pentru aceasta trebuie să stocăm date despre instruiţi şi instructori. Pentru
3	fiecare participant la curs (în jur de 5000), identificaţi prin cod, vrem să stocăm
4	codul numeric personal, numele, vârsta, sexul, locul naşterii, numele
5	angajatorului, adresa şi numărul de telefon, angajatorii anteriori (şi perioada
6	angajării), cursurile urmate (există circa 200 de cursuri) şi aprecierea finală la
7	fiecare curs. Avem nevoie de asemenea să reprezentăm seminariile la care
8	fiecare participant este aşteptat în prezent şi, pentru fiecare zi, locurile şi orele la
9	care clasele sunt ocupate.
10	Fiecare curs are un cod şi un titlu şi fiecare curs poate fi organizat de oricâte ori.
11	Fiecărei organizări a unui curs particular îi spunem „ediţie” a cursului. Pentru
12	fiecare ediţie vom reprezenta data de start, data de sfârşit şi numărul
13	participanţilor. Dacă un instruit este dintr-o profesie liberală trebuie cunoscut
14	domeniul de expertiză şi dacă este necesar titlul său. Pentru oricine care lucrează
15	la companie, vom stoca nivelul şi poziţia deţinută.
16	Pentru fiecare instructor (circa 300) vom preciza numele, vârsta, locul naşterii,
17	ediţia cursului predat, cursurile predate în trecut şi cursurile pe care un titular este
18	calificat să le ţină.
19	Se stochează numerele de telefon ale tuturor instructorilor.
20	Un instructor poate fi angajat permanent al companiei de training sau poate fi
21	angajat temporar.

Este evident că cerinţele au ambiguităţi.

Spre exemplu există:

- *participanţi* sau *instruiţi*
- *titulari* sau *instructori*
- *cursuri* sau *seminarii*

Reguli pentru scrierea specificaţiilor mai precis şi fără ambiguităţi:

- Se alege un nivel potrivit de abstractizare. Se evită termenii prea generali sau prea specifici.

  Exemplu:

  perioadă (linia 5) - *dată start* şi *dată sfârşit*

  titlu (linia 14) - *titlu profesional*

  apreciere (linia 6) - *notă*

- Se standardizează structura propoziţiei.

  Exemplu:

  „pentru <*concept*> păstrăm <*proprietăţi*>”

- Se evită frazele complexe

  Exemplu:

  angajat este preferat lui *oricine care lucrează pentru o companie*

- Se identifică sinonimele şi omonimele

  Exemplu:

  titular şi *instructor*

  participant curs şi *instruit*

  loc care înseamnă *locul de naştere* cât şi *locul unde se ţin orele*

  Pentru sinonime se foloseşte un singur termen iar pentru omonime se caută alţi termeni.

- Se marchează explicit referinţele. Absenţa referinţelor dintre termeni duce la concepte ambigue.

  Exemplu:

  la linia 5 *adresa* şi *numărul de telefon* se referă la angajat sau angajator ?

- Se construieşte un vocabular. Pentru fiecare termen, vocabularul conţine:

  - o scurtă descriere

  - sinonime posibile

  - referinţe la alţi termeni conţinuţi de vocabular cu care este în legătură logică

Termen	Descriere	Sinonime	Legături
Instruit	Participant la curs. Poate fi angajat sau să aibă o profesie liberală.	Participant	Curs, Companie
Instructor	Titular curs. Poate fi angajat temporar.	Titular	Curs
Curs	Curs oferit. Poate avea mai multe ediţii	Seminar	Instructor, Instruit
Companie	Compania unde este angjat participantul sau unde a fost angajat.		Instruit

Figura 2. Exemplu de vocabular

Se pot rescrie specificaţiile şi se grupează cerinţele ca în figura următoare.

Cerinţe generale
Se doreşte crearea unei baze de date pentru o companie care derulează cursuri de instruire. Se doreşte păstrarea datelor pentru instruiţi şi instructori.
Cerinţe referitoare la instruiţi
Pentru fiecare instruit (în jur de 5000), identificat de un cod, se va păstra codul numeric personal, nume, vârstă, sex, oraşul de naştere, angajatorul curent, angajatorul precedent (cu data de start şi data de sfârşit a perioadei în care a fost angajat), ediţiile cursurilor pe care instruitul le urmează în prezent şi cele pe care le-a urmat împreună cu nota obţinută.
Cerinţe referitoare la angajatorii instruiţilor
Pentru fiecare angajator a unui instruit se va păstra numele, adresa şi numărul de telefon
Cerinţe referitoare la cursuri
Pentru fiecare curs (în jur de 200) trebuie stocat numele şi codul. Fiecare organizare a unui curs anume se numeşte ediţie a cursului. Pentru fiecare ediţie se va stoca data de start, data de sfârşit şi numărul participanţilor. Pentru ediţia curentă se vor păstra datele, sălile de clasă, şi momentele în care clasa este ocupată.
Cerinţe referitoare la tipuri specifice de instruiţi
Pentru un instruit care practică o profesie liberală (liber profesionist), se va păstra domeniul de expertiză şi eventual titlul profesional. Pentru un instruit care este angajat, se vor păstra nivelul şi poziţia ocupată.
Cerinţe referitoare la instructori
Pentru fiecare instructor (în jur de 300), se vor păstra numele, vârsta, oraş de naştere toate numerele de telefon, ediţiile cursurilor predate în prezent şi în trecut, şi cursurile pe care calificat să le predea. Instructorii pot fi angajaţi permanenţi ai companiei de training sau pot fi angajaţi temporar

Figura 3. Exemplu de structurare a cerinţelor

După specificarea datelor trebuie specificate *operaţiile* care trebuie executate asupra acestor date.

În cazul nostru operaţiile ar putea fi:

- Operaţia 1: se inserează un nou instruit incluzând datele despre el (se realizează de aproximativ 40 de ori pe zi)
- Operaţia 2: se atribuie un instruit unei ediţii a unui curs (de 50 de ori pe zi)
- Operaţia 3: se inserează un nou instructor, incluzând toate datele şi cursurile pe care acesta este calificat să le predea
- Operaţia 4: se atribuie un instructor calificat pentru o ediţie a unui curs (de 15 ori pe zi)
- Operaţia 5: se afişează toate informaţiile despre o ediţie anterioară a cursului: titlu, orar, număr de instruiţi (de 10 ori pe zi)
- Operaţia 6: afişează toate cursurile disponibile, cu informaţii despre instructorii calificaţi să le ţină (de 20 de ori pe zi)
- Operaţia 7: pentru fiecare instructor, se găsesc instruiţii pentru toate cursurile pe care le ţine sau le-a ţinut(de 5 ori pe săptămână)
- Operaţia 8: se realizează o analiză statistică a tuturor instruiţilor cu toate informaţiile despre ei, despre ediţiile cursurilor pe care le-au urmat şi notele obţinute (de 10 ori pe lună)

7.2 Strategii de proiectare

Strategia top-down (se sus în jos)

- Schema conceptuală este obţinută printr-o serie de rafinări succesive ale schemei iniţiale ce descrie toate cerinţele prin intermediul câtorva concepte abstracte.
- Fiecare nivel reprezentat conţine o schemă ce descrie informaţii diverse la diferite grade de detaliu.
- Trecerea de la un nivel la altul se face cu ajutorul unor transformări numite *primitive de transformare de sus în jos*
  - operează pe un singur concept al schemei
  - îl transformă într-o structură de complexitate mai ridicată, capabilă să descrie conceptul iniţial în detaliu
  - sunt disponibile 6 primitive de transformare

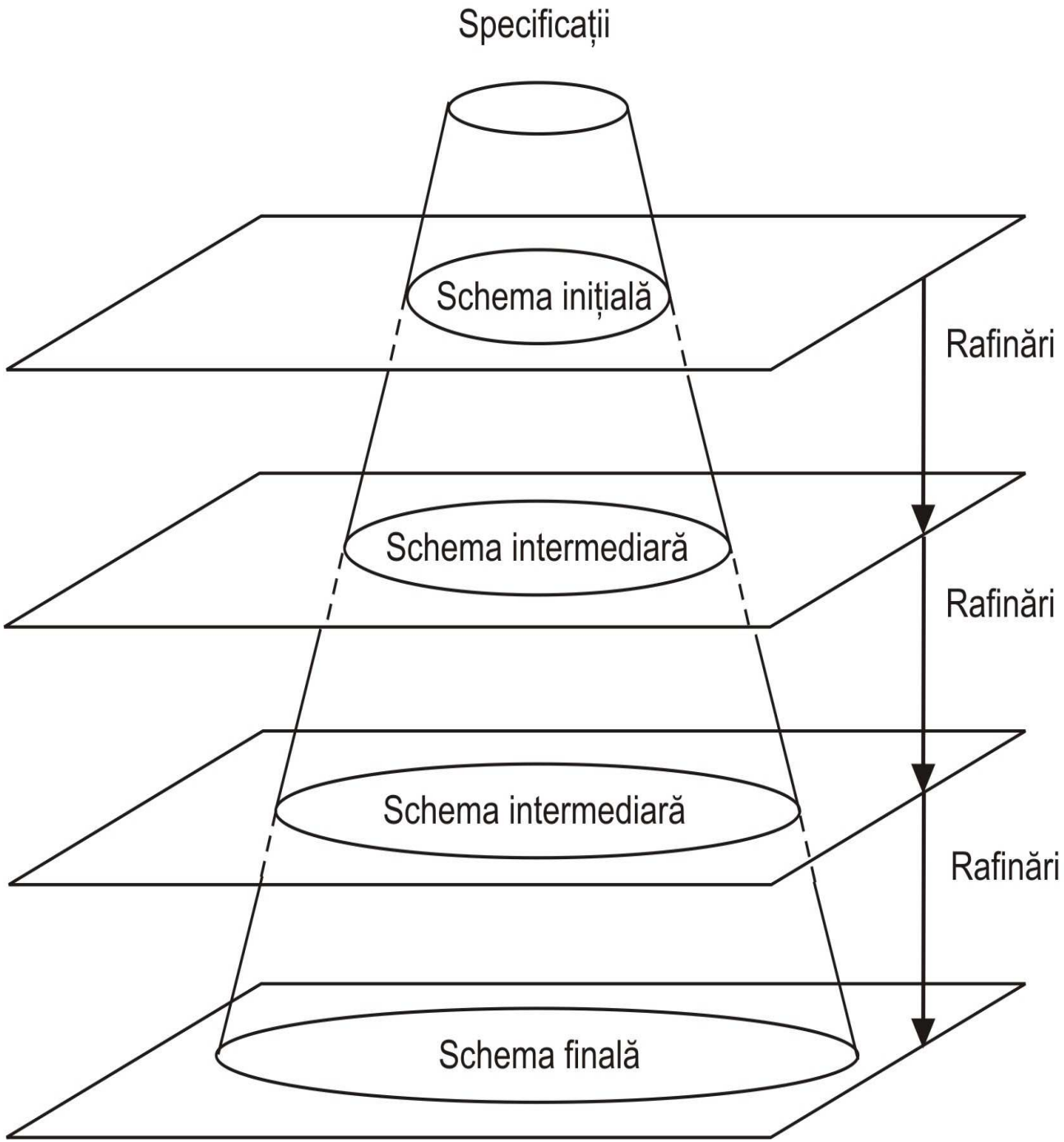
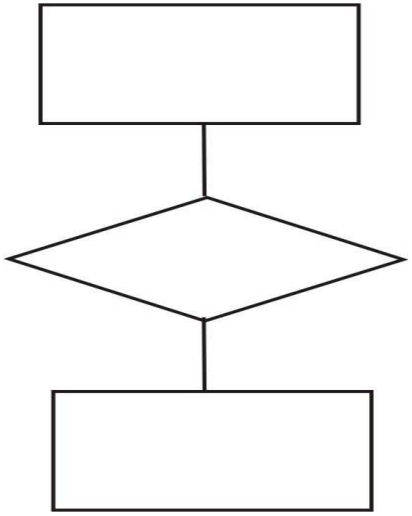


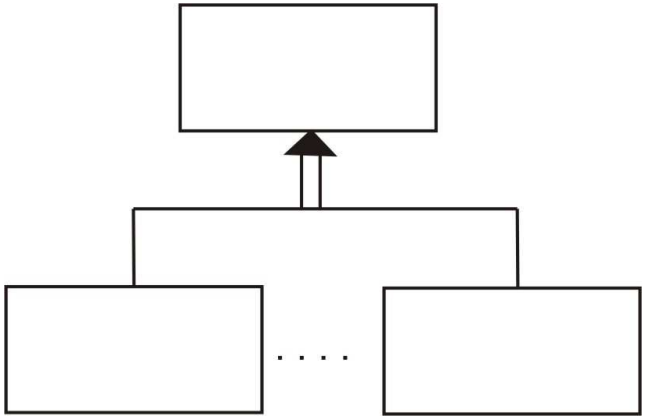
Figura 4. Strategia top-down

Transformare	Concept iniţial	Rezultat
T_1 De la o entitate la două entităţi şi relaţia dintre ele		

- se aplică atunci când o entitate descrie două concepte logice diferite legate unele de altele

Exemplu

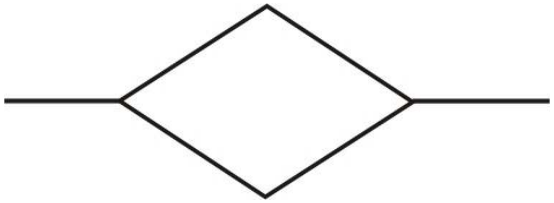
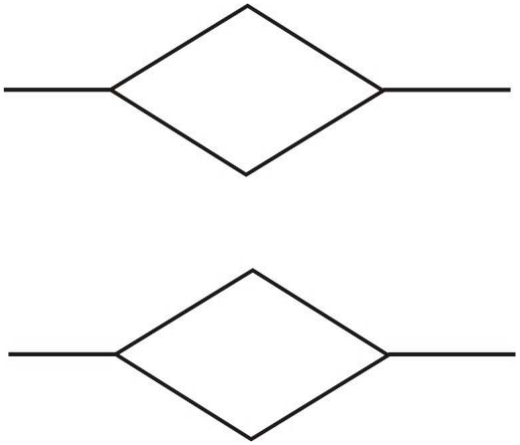
- în aplicaţia descrisă în secţiunea anterioară se poate începe cu entitatea CURS
- acest concept pare prea abstract, putând face deosebirea între:
  - TIPCURS (care are un cod şi un titlu)
  - EDIŢIECURS (care are o dată de start şi una de sfârşit)
- aceste două entităţi pot fi legate prin relaţia TIP

Transformare	Concept iniţial	Rezultat
T_2 De la o entitate la o generalizare		

- este aplicată atunci când o entitate este alcătuită din sub-entităţi

Exemplu

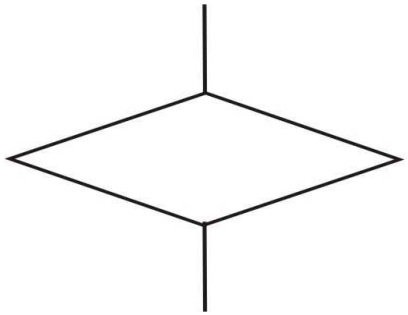
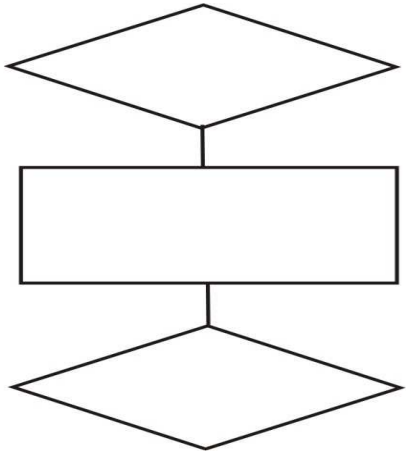
- în aplicaţia noastră această transformare are loc atunci când ne dăm seama că printre cei instruiţi se poate distinge între:
  - ANGAJAT
  - LIBERPROFESIONIST

Transformare	Concept iniţial	Rezultat
T_3 De la o relaţie la relaţii multiple		

- se aplică atunci când o relaţie descrie două sau mai multe concepte diferite legând aceleaşi entităţi

Exemplu

  - în relaţia PREDARE între instructori şi cursuri, PREDARECURENTĂ poate fi separată de PREDAREANTERIOARĂ

Transformare	Concept iniţial	Rezultat
T_4 De la o relaţie la o entitate cu relaţii		

- se aplică atunci când o relaţie descrie un concept cu existenţă autonomă

Exemplu

  - dacă relaţia CONTRACT între o entitate CONSULTANT şi o entitate COMPANIE are multe atribute, atunci ea este mai bine reprezentată printr-o entitate legată de altele prin intermediul unor relaţii binare

Transformare	Concept iniţial	Rezultat
T_5 Adăugarea atributelor la o entitate		

- se aplică pentru adăugarea unor proprietăţi (atribute) entităţilor.

Exemplu

- atunci când rafinăm entitatea INSTRUIT prin adăugarea atributelor:

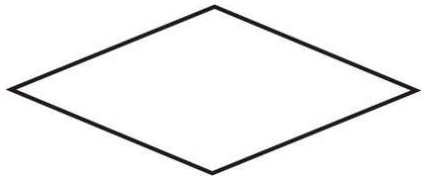
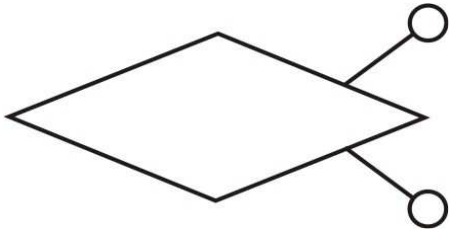
  CNP

  Nume

  Vârstă

  Sex

  OraşDeNaştere

Transformare	Concept iniţial	Rezultat
T_6 Adăugarea atributelor la o relaţie		

- se aplică atunci când se adaugă proprietăţi la o relaţie, într-o manieră similară transformării T_5

Avantajul strategiei top - down

proiectantul poate începe cu o reprezentare completă a cerinţelor, chiar dacă lipsesc unele detalii

Dezavantajul

este necesară o viziune globală asupra tuturor conceptelor, ceea ce este dificil de realizat în cazul aplicaţiilor complexe

Strategia bottom-up (de jos în sus)

- specificaţiile iniţiale sunt descompuse în componente până când fiecare componentă descrie un fragment elementar al specificaţiilor
- în acest punct, componentele sunt reprezentate prin scheme conceptuale simple
- aceste scheme vor fi combinate pentru a se obţine schema finală
- şi în acest caz se utilizează transformări elementare - *primitive de transformare de jos în sus*
- aceste primitive introduc în schemă concepte noi care nu au fost prezente până în acel moment, capabile să descrie aspecte ale aplicaţiei care nu au fost luate în considerare

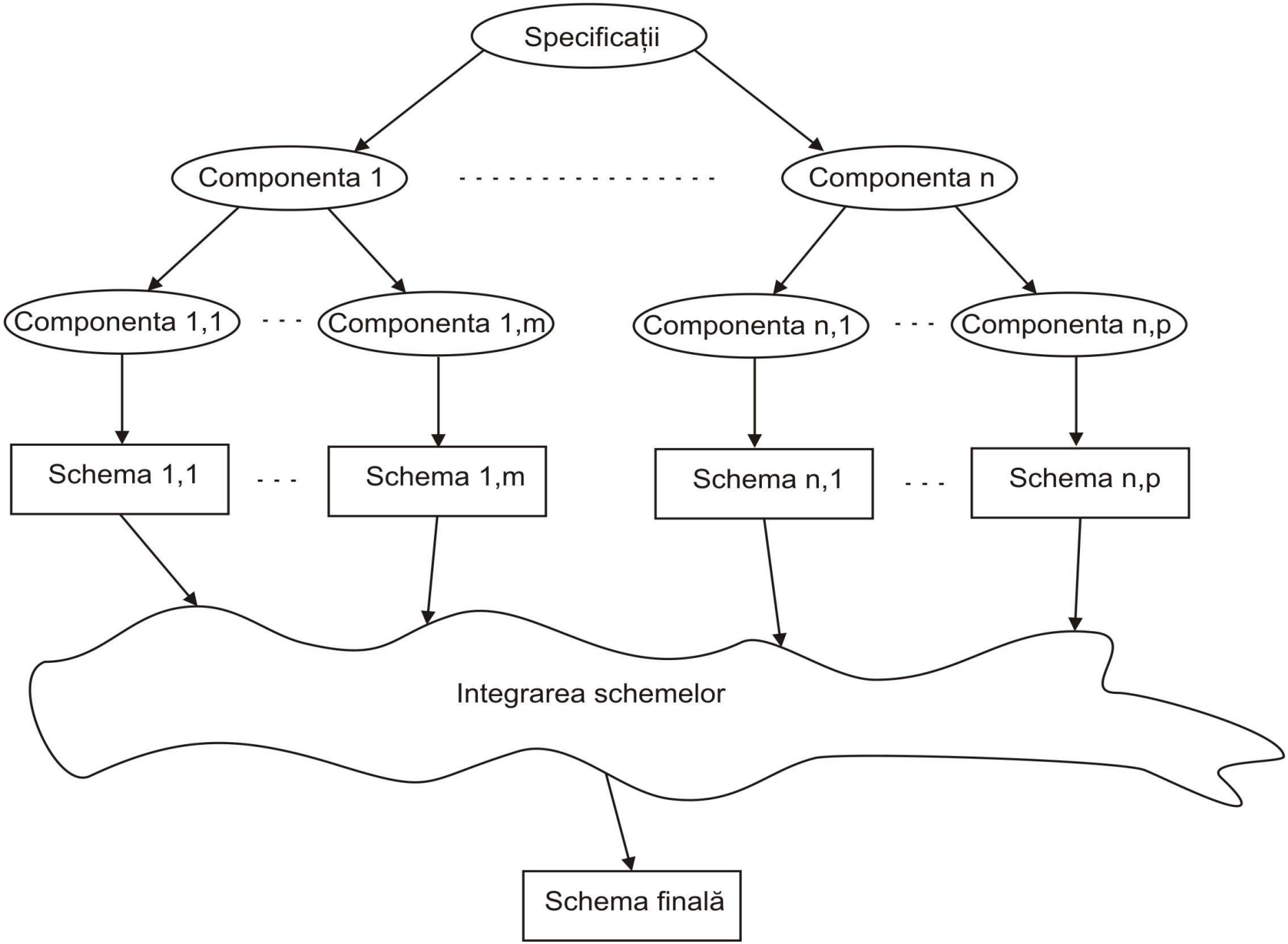


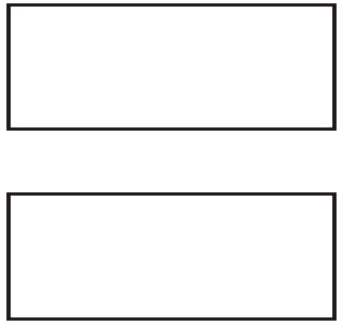
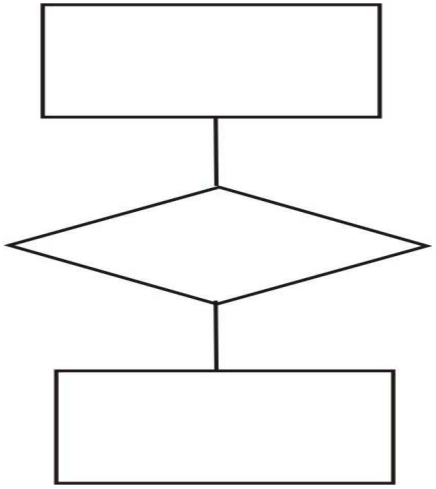
Figura 6. Strategia bottom-up

Transformare	Concept iniţial	Rezultat
T_1 Generarea unei entităţi		

- se aplică atunci când se identifică în specificaţii o clasă de obiecte cu proprietăţi comune

Exemplu

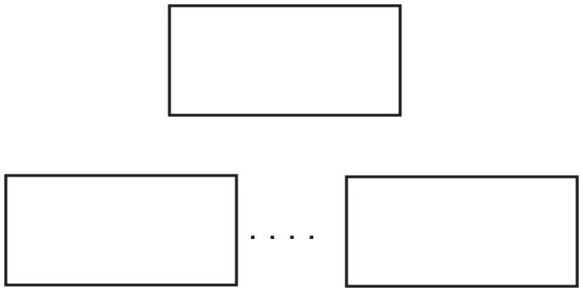
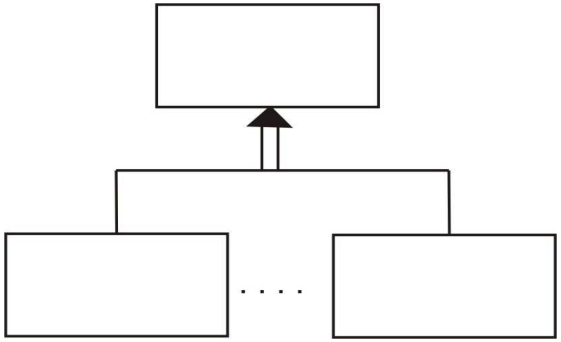
- în aplicaţia descrisă anterior se poate identifica entitatea CLASĂ (ce păstrează o anumită clasă la un anumit moment)

Transformare	Concept iniţial	Rezultat
T_2 Generarea unei relaţii		

- se aplică atunci când se identifică în specificaţii o legătură logică între două entităţi

Exemplu

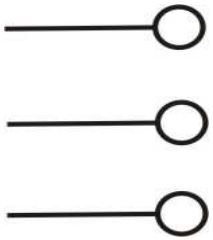
- în aplicaţia noastră se poate identifica relaţia CALIFICARE între entităţile INSTRUCTOR şi CURS

Transformare	Concept iniţial	Rezultat
T_3 Generarea unei generalizări		

- se aplică atunci când se identifică în specificaţii o generalizare între entităţi

Exemplu

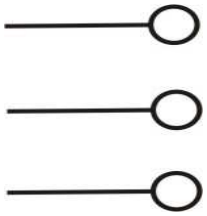
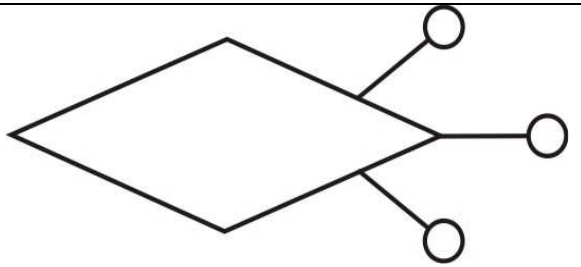
- entitatea INSTRUCTOR este o generalizare a entităţilor PERMANENT şi TEMPORAR

Transformare	Concept iniţial	Rezultat
T_4 Agregarea unor atribute pe o entitate		

- se aplică atunci când se identifică în specificaţii o entitate care poate fi privită ca o agregare a unor serii de atribute

Exemplu

- se identifică entitatea INSTRUIT cu proprietăţile
  - CNP
  - Nume
  - Vârstă
  - Sex
  - OraşDeNaştere

Transformare	Concept iniţial	Rezultat
T_5 Agregarea unor atribute pe o relaţie		

- se aplică atunci când o relaţie poate fi privită ca o agregare a unor atribute

Avantajul strategiei bottom – up

permite descompunerea problemei în componente simple care pot fi uşor identificate şi astfel procesul de proiectare poate fi atribuit mai multor proiectanţi dacă este necesar

Dezavantajul

este necesară integrarea mai multor scheme conceptuale

Strategia inside-out (din interior spre exterior)

- poate fi privită ca o particularizare a strategiei de jos în sus
- se începe cu câteva concepte importante şi apoi pe baza acestora, proiectarea se extinde radial
- cu alte cuvinte se reprezintă mai întâi conceptele cele mai apropiate de conceptele iniţiale şi apoi procesul de proiectare se mută spre conceptele mai depărtate prin intermediul navigării prin specificaţii
- avantajul acestei strategii constă în eliminarea paşilor de integrare din strategia de jos în sus
- este necesară examinarea, din timp în timp, a tuturor specificaţiilor căutând concepte ce nu au fost reprezentate încă

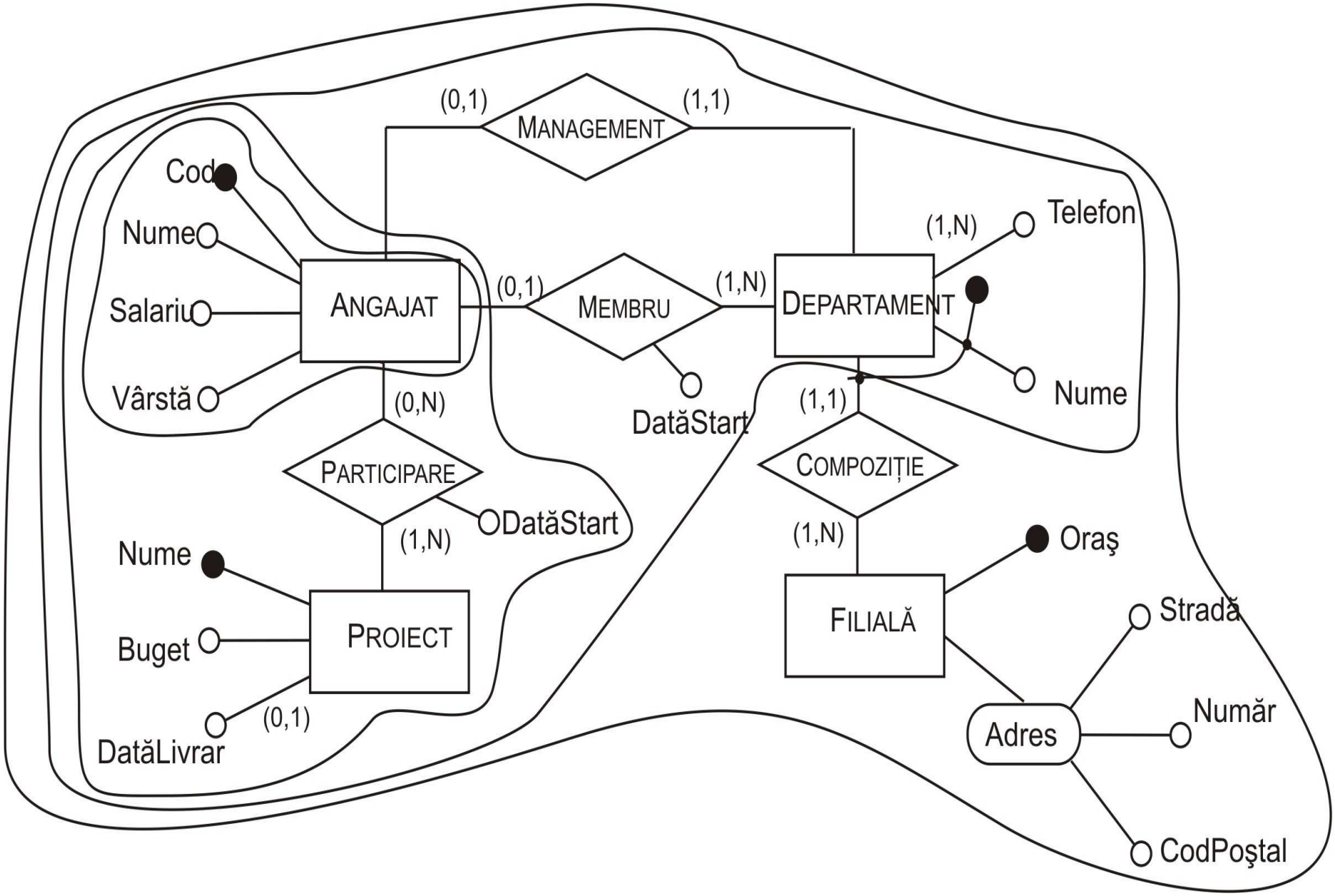


Figura 8. Exemplu de strategie inside-out

Ariile indicate prezintă o dezvoltare cronologică posibilă a schemei.

Strategia mixtă

- se poate adopta o strategie mixtă care combină avantajele strategiilor top-down, bottom-up şi inside-out
- proiectanţii descompun cerinţele în componente conform strategiei bottom-up, dar nu se dezvoltă componentele separat
- în acelaşi timp se defineşte o *schemă cadru* care conţine, la nivel abstract, principalele componente ale aplicaţiei
- schema cadru furnizează o viziune sintetică asupra procesului de proiectare şi uşurează integrarea schemelor dezvoltate separat

Pentru exemplul prezentat în secţiunile anterioare o schemă cadru posibilă este prezentată în figura următoare:

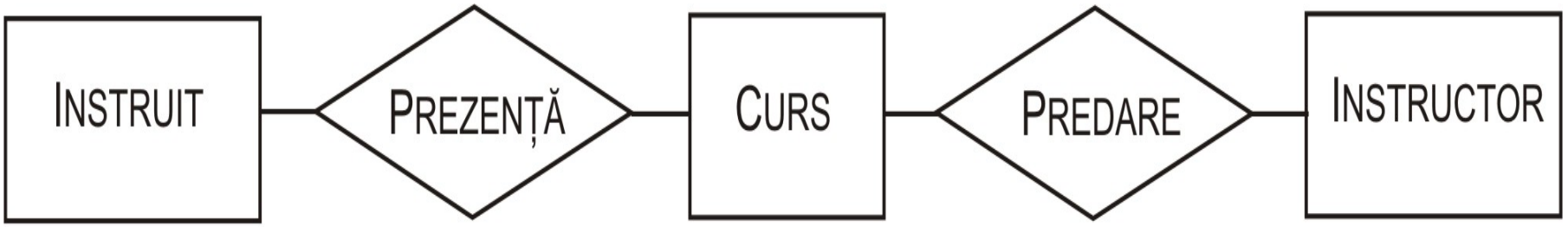


Figura 9. Schema cadru pentru procesul de instruire într-o companie

Din acest punct se pot examina separat conceptele principale prin:

- rafinări graduale (urmând paşii strategiei top-down)

sau

- se pot extinde componentele cu concepte care nu au fost încă reprezentate (conform strategiei bottom-up)

7.3 Calitatea unei scheme conceptuale

Proprietăţile utilizate pentru a stabili calitatea unei scheme sunt:

Corectitudinea schemei

O schemă conceptuală este corectă dacă utilizează corect construcţiile puse la dispoziţie de modelul conceptual.

Se pot defini două tipuri de erori:

- *erorile sintactice* marchează utilizarea ilegală a unei construcţii (ex.: generalizarea dintre relaţii în detrimentul entităţilor)
- *erorile semantice* marchează utilizarea unei construcţii care nu-şi urmăreşte definiţia (ex.: utilizarea unei relaţii pentru a descrie faptul că o entitate este o specializare a altei entităţi)

Caracterul complet al schemei

O schemă conceptuală este completă dacă include concepte ce reprezintă toate cerinţele de date şi care permit execuţia tuturor operaţiilor incluse în cerinţele operaţionale.

Accesibilitatea schemei

O schemă conceptuală este accesibilă când reprezintă cerinţele într-un mod natural şi uşor de înţeles.

Minimalitatea schemei

- schema este minimală când toate specificaţiile datelor sunt reprezentate doar o singură dată în schemă
- schema nu este minimală când apar *redundanţele* – concepte derivate din altele
  - o sursă tipică de redundanţe este prezenţa ciclurilor determinate de prezenţa relaţiilor şi/sau generalizărilor
  - câteodată redundanţele sunt necesare din motive de proiectare, aceste situaţii fiind precizate în documentaţie

7.4 Metodă de abordare a proiectării conceptuale

În practică se aplică foarte rar o singură strategie de proiectare conceptuală.

Independent de strategia aleasă, apare necesitatea modificării schemei utilizând:

- transformări top-down (prin care se rafinează conceptele deja prezentate)
- transformări bottom-up (prin care se adaugă concepte noi)

Etapele ce trebuie parcurse pentru realizarea unei scheme conceptuale sunt:

1. *Analiza cerinţelor*
   - Construirea unui vocabular
   - Analiza cerinţelor şi eliminarea ambiguităţilor
   - Gruparea cerinţelor

2. *Etapa de bază*

   - Identificarea celor mai relevante concepte şi reprezentarea lor într-o schemă cadru

3. *Descompunerea* (folosită dacă este potrivită sau necesară)

   - Descompunerea cerinţelor cu referire la conceptele prezentate în schema cadru

4. *Etapa iterativă* (se repetă pentru toate schemele până când fiecare specificaţie este reprezentată)

   - Rafinarea conceptelor pe baza cerinţelor
   - Adăugarea de concepte noi care descriu părţi ale cerinţelor nereprezentate încă

5. *Integrarea*

   - Integrarea sub-schemelor într-o schemă generală ţinând cont de schema cadru

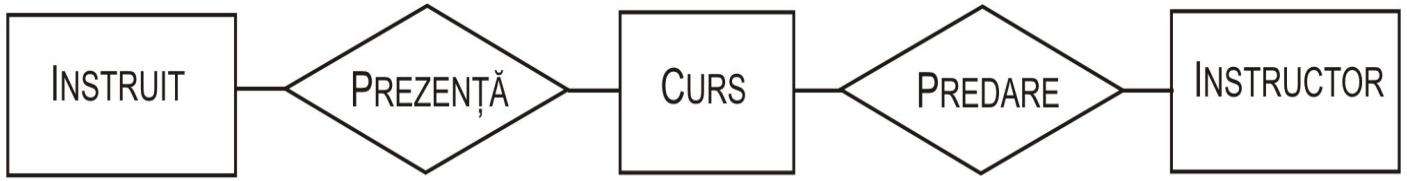
6. *Analiza calităţii*

- Verificarea corectitudinii şi realizarea restructurărilor necesare
- Verificarea caracterului complet al schemei şi realizarea restructurărilor necesare
- Verificarea minimalităţii, listarea redundanţelor şi dacă este necesar realizarea restructurărilor necesare
- Verificarea accesibilităţii şi realizarea restructurărilor necesare dacă este necesar

7.5 Exemplu de proiectare conceptuală

Se consideră exemplul unui proces de instruire din cadrul unei companii despre care am mai discutat şi în secţiunile anterioare.

Schema cadru



Din acest punct se poate decide analizarea separată a specificaţiilor pentru instruiţi, cursuri şi instructori şi de a aplica o strategie inside-out pentru fiecare.

Instruiţi

Se pot identifica două tipuri:

- *angajaţi*
- *liber profesionişti*

Aceste entităţi se reprezintă ca specializări ale entităţii INSTRUIT; generalizarea este totală.

Este necesară reprezentarea *angajatorilor instruiţilor*. Aceasta se poate face introducând entitatea ANGAJATOR care este legată printr-o relaţie de ANGAJAT.

Este necesară de asemenea distincţia între conceptele angajare actuală şi anterioară.

Decidem să divizăm relaţia în două relaţii: ANGAJAREANTERIOARĂ şi ANGAJAREACTUALĂ.

Prima are o dată de start şi una de sfârşit şi este legată de entitatea INSTRUIT (deoarece şi liber profesioniştii se poate să fi fost angajaţi).

A doua relaţie are doar dată de start şi este legată de entitatea ANGAJAT.

Adăugând atribute entităţilor şi relaţiilor, cardinalităţi pentru relaţii şi identificatori ai entităţilor se obţine schema din figura 10.

Se observă că entitatea INSTRUIT are doi identificatori (*Cod* şi *CNP*). Atributul *TitluProfesional* este opţional.

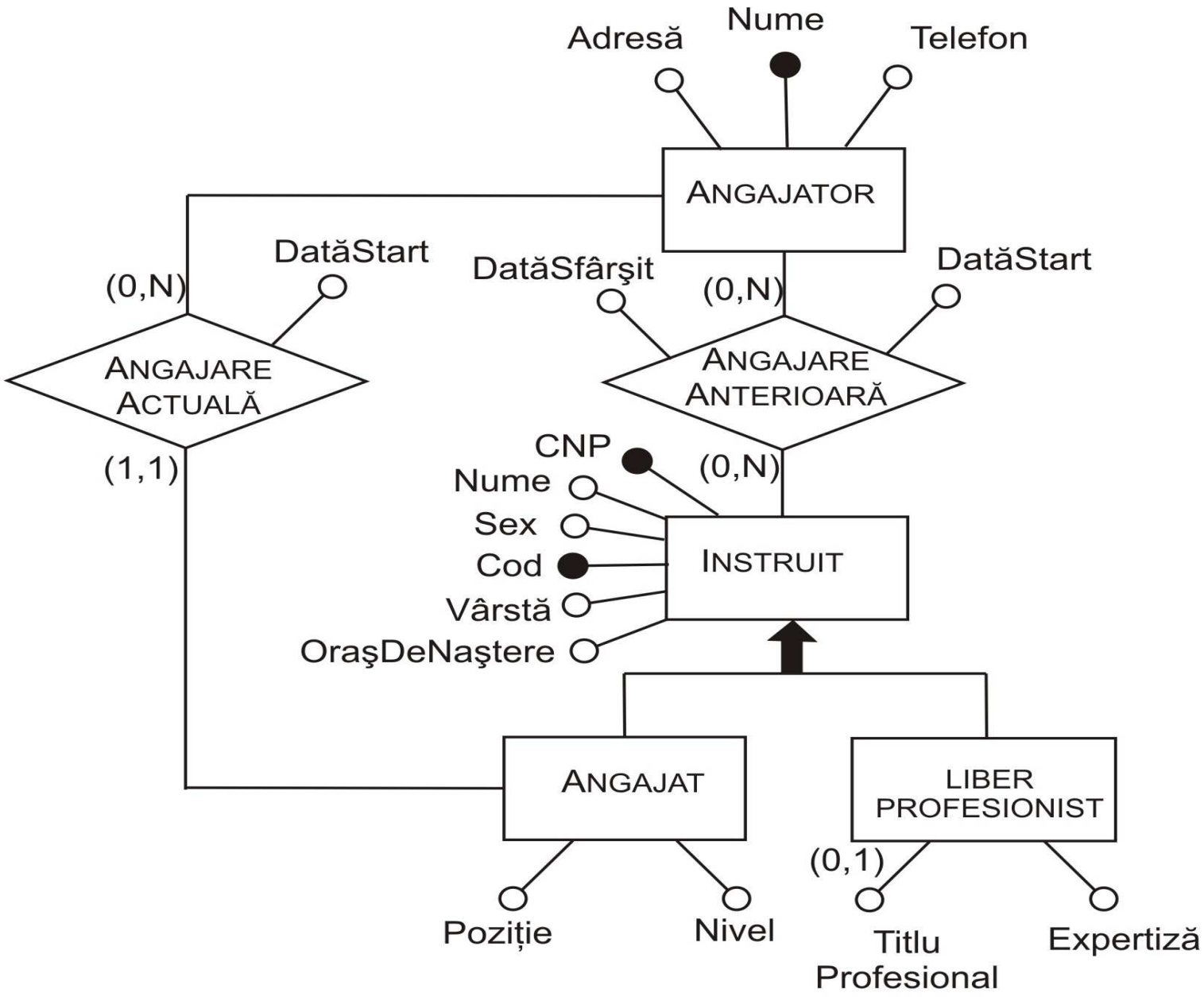


Figura 10. Rafinarea unei porţiuni a schemei cadru

Instructori

Se disting cazurile în care aceştia sunt fie *angajaţi permanenţi* ai companiei, fie *angajaţi temporar*. Se realizează astfel o generalizare totală, cu entitatea părinte INSTRUCTOR.

Se adaugă atributele precizate în specificaţii: Nume, Vârstă, OraşDeNaştere şi Telefon.

Se observă cu nu se poate stabili un identificator pe baza acestor atribute $\Rightarrow$ se decide folosirea CNP-ului instructorului chiar dacă nu este cerut în specificaţii.

Schema rezultată este prezentată în figura 11.

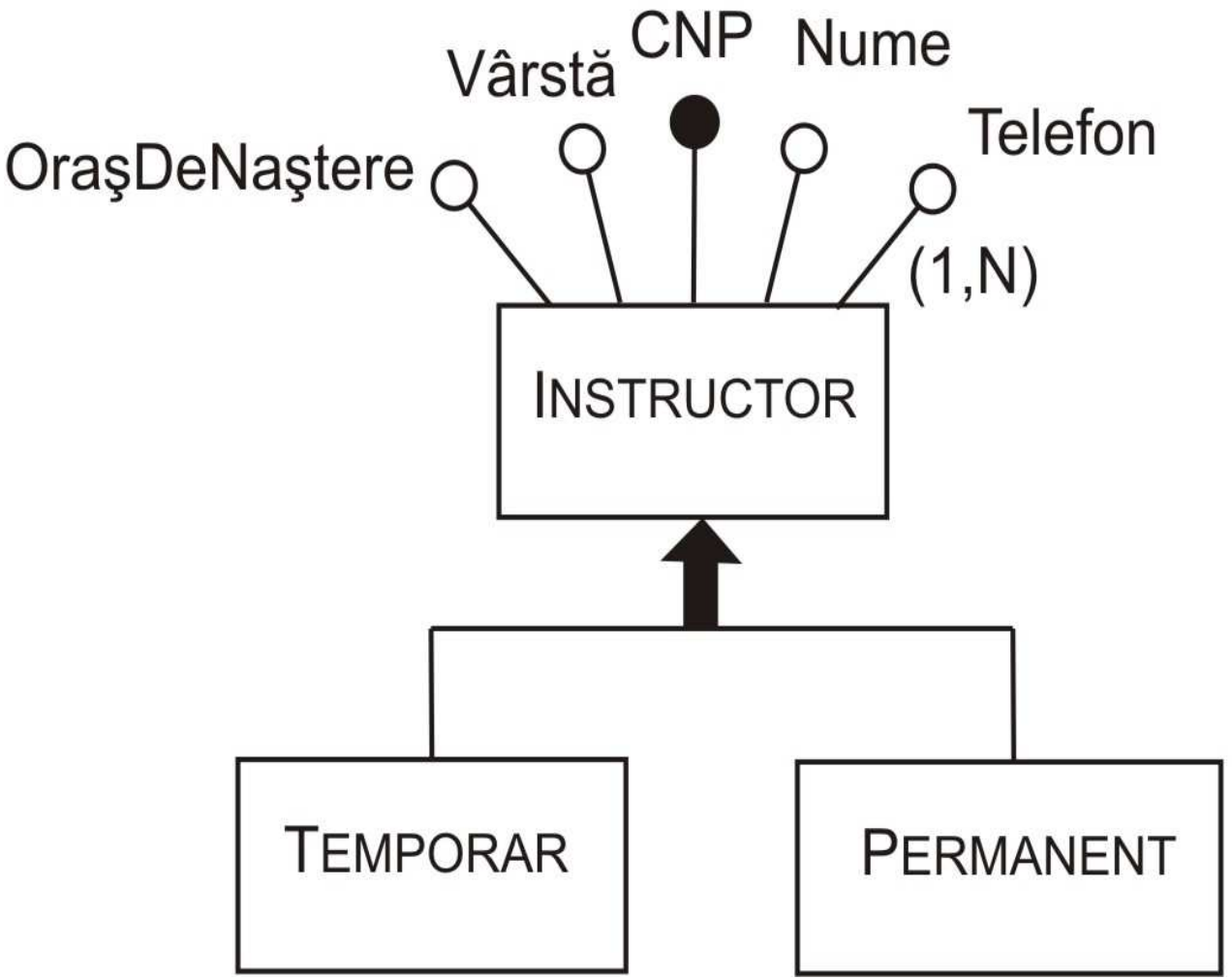


Figura 11. Rafinarea unei alte porţiuni a schemei cadru

În ceea ce priveşte entitatea CURS, există două concepte distincte legate:

- un concept abstract al cursului (cu *nume* şi *cod*)
- ediţia cursului (cu *dată de start*, *dată de sfârşit* şi numărul de participanţi)

Vom reprezenta aceste concepte cu două entităţi distincte legate prin relaţia TIP.

Clasele unui curs se pot descrie printr-o entitate legată de ediţiile cursurilor prin relaţia COMPOZIŢIE.

Se adaugă apoi atribute, cardinalităţi şi identificatori.

O clasă este identificată prin *sală*, *timp* şi *dată*.

Pentru ediţiile unui curs, presupunem că două ediţii ale aceluiaşi curs nu pot începe în aceeaşi zi şi astfel un identificator pentru entitatea EDIŢIECURS este format din atributul *DatăStart* şi entitatea CURS.

Schema rezultată este prezentată în figura 12.

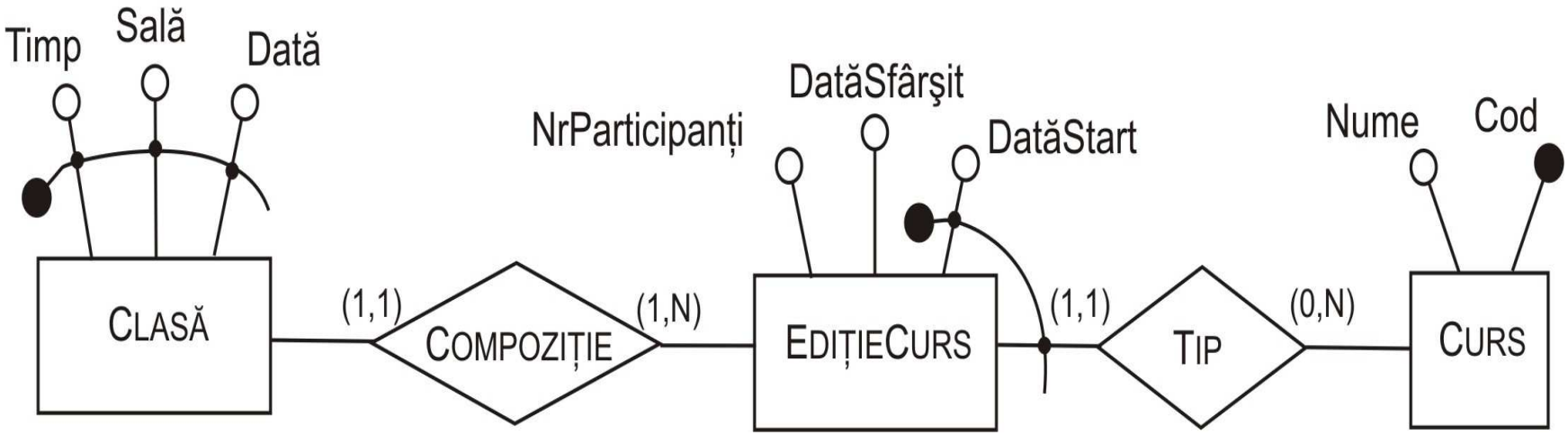


Figura 12. Rafinarea unei alte părţi a schemei cadru

Schema finală este obţinută prin integrarea schemelor obţinute până în acest punct.

Vom începe cu schemele referitoare la instructori şi cursuri.

În schema cadru aceste părţi sunt legate prin relaţia PREDARE.

Această relaţie trebuie rafinată.

Se identifică trei legături diferite între instructori şi cursuri: predare curentă, predare anterioară şi calificare.

Aceste legături se reprezintă prin intermediul a trei relaţii:

- primele două leagă entităţile INSTRUCTOR şi EDIŢIECURS
- a treia leagă INSTRUCTOR de CURS

În aceste moment se poate face integrarea.

Ţinând cont de schema cadru, trebuie clarificată relaţia între cursuri şi instruiţi.

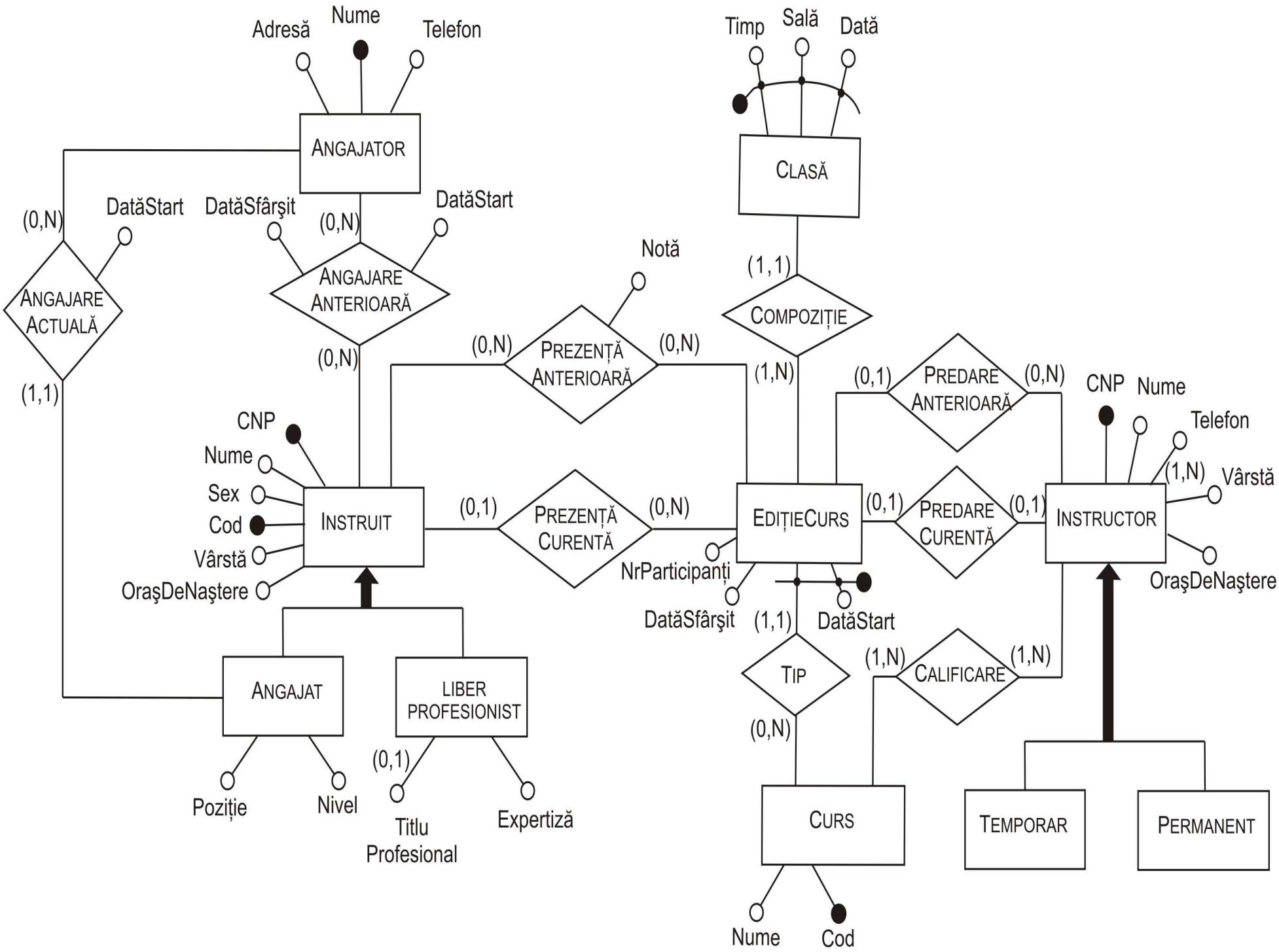
Apar două cazuri:

- prezenţă curentă
- prezenţă anterioară

$\Rightarrow$ definim două relaţii între entităţile INSTRUIT şi EDIŢIECURS.

Pentru o prezenţă anterioară interesează nota finală.

Se obţine în final schema din figura 13.


Adresă
Nume
Telefon
ANGAJATOR
(0,N)
DataStart
ANGAJARE ACTUALĂ
(1,1)
DataSfârşit
(0,N)
DataStart
ANGAJARE ANTERIOARĂ
(0,N)
Timp
Sală
Dată
CLASĂ
(1,1)
COMPOZIȚIE
(1,N)
Notă
(0,N)
PREZENȚĂ ANTERIOARĂ
(0,N)
(0,1)
PREDARE ANTERIOARĂ
(0,N)
CNP
Nume
Sex
Cod
Vârstă
OraşDeNaştere
INSTRUIT
(0,1)
PREZENȚĂ CURENTĂ
(0,N)
EDIȚIECURS
NrParticipanți
DataSfârşit
(1,1)
DataStart
(0,1)
PREDARE CURENTĂ
(0,1)
INSTRUCTOR
CNP
Nume
Telefon
(1,N)
Vârstă
OraşDeNaştere
ANGAJAT
Poziție
Nivel
LIBER PROFESIONIST
(0,1)
Titlu Profesional
Expertiză
TIP
(0,N)
(1,N)
CALIFICARE
(1,N)
CURS
Nume
Cod
TEMPORAR
PERMANENT

În acest moment se începe verificarea schemei obţinute.

Se verifică dacă schema este completă prin întoarcerea la specificaţii şi verificarea dacă toate datele sunt reprezentate şi toate operaţiile pot fi efectuate.

Exemplu

Să considerăm operaţia 7, în care se cer instruiţii pentru toate cursurile ţinute de un instructor.

Datele pentru această operaţie se găsesc pe schemă în felul următor:

- se pleacă de la entitatea INSTRUCTOR
- se trece prin relaţiile PREDARECURENTĂ şi PREDAREANTERIOARĂ, entitatea EDIŢIECURS, relaţiile PREZENŢĂCURENTĂ şi PREZENŢĂANTERIOARĂ şi apoi se ajunge la entitatea INSTRUIT

Cu privire la minimalitate, să notăm că există o redundanţă în schemă:

- atributul *NrParticipanţi* al entităţii EDIŢIECURS se poate obţine prin numărarea numărului de instanţe ale entităţii INSTRUIT care sunt legate de ediţia respectivă.
- se va discuta despre eliminarea sau menţinerea acestei redundanţe în capitolul următor, referitor la proiectarea logică

Trebuie menţionat în final că schema trebuie să aibă o documentaţie potrivită.

- este importantă descrierea restricţiilor posibile care nu sunt exprimate în schemă, sub forma regulilor de operare
- *Exemplu*: un instructor predă un curs doar dacă este calificat să o facă